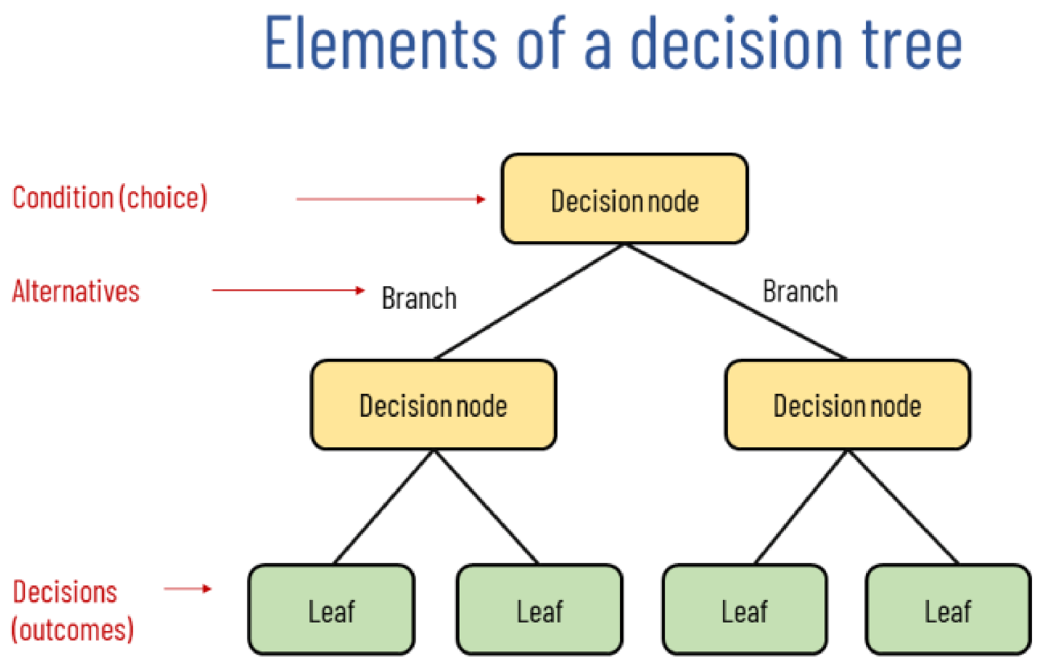
CT-562
MACHINE LEARNING

NED University of Engineering & Technology

THIS LECTURE

- Decision Tree Classification Algorithm



DECISION TREE CLASSIFICATION ALGORITHM

DECISION TREE CLASSIFICATION ALGORITHM

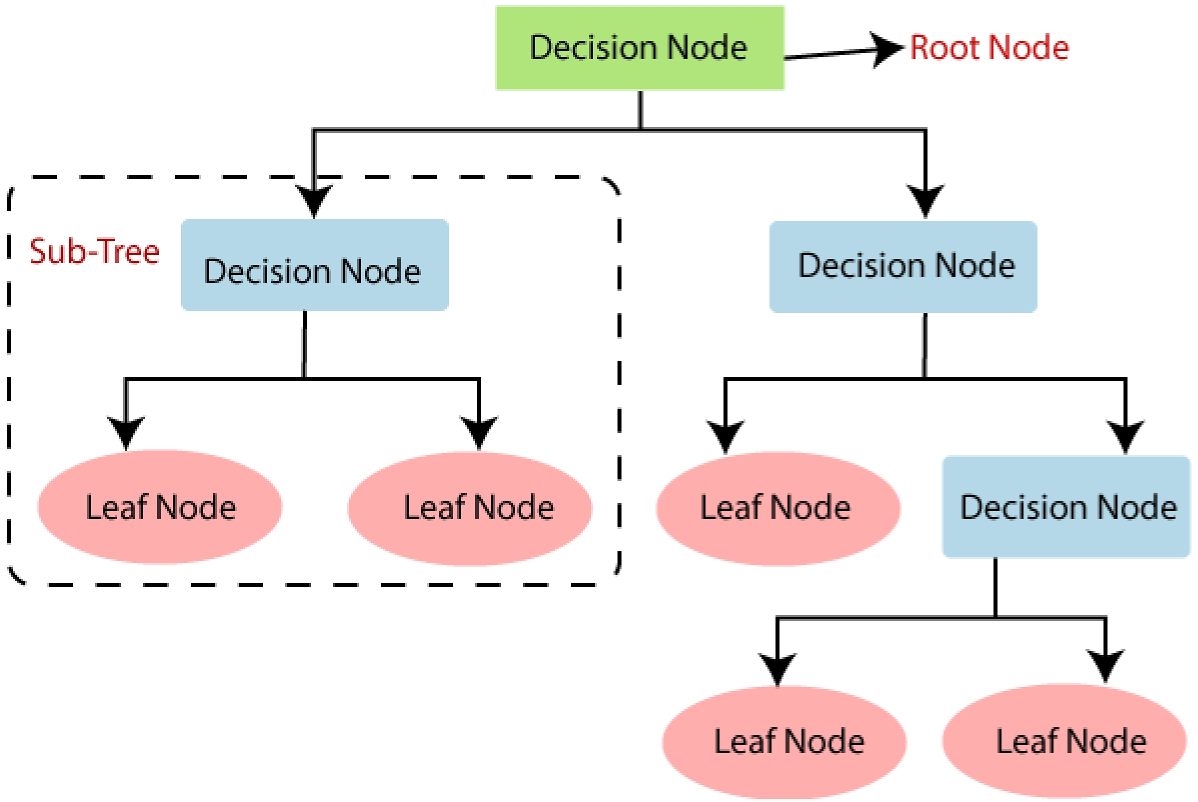
- It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome.
- In a Decision tree, there are two nodes, which are the **Decision Node** and **Leaf Node**. Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.

DECISION TREE CLASSIFICATION ALGORITHM

- It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.
- It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.
- In order to build a tree, we use the **CART** algorithm, which stands for **Classification and Regression Tree algorithm**.
- A decision tree simply asks a question, and based on the answer (**Yes/No**), it further split the tree into subtrees.

DECISION TREE CLASSIFICATION ALGORITHM

- General structure of a decision tree



DECISION TREE CLASSIFICATION ALGORITHM

- There are various algorithms in Machine learning, so choosing the best algorithm for the given dataset and problem is the main point to remember while creating a machine learning model. Below are the two reasons for using the Decision tree:
  - Decision Trees usually mimic human thinking ability while making a decision, so it is easy to understand.
  - The logic behind the decision tree can be easily understood because it shows a tree-like structure

DECISION TREE TERMINOLOGIES

- **Root Node**: The root node is from where the decision tree starts. It represents the entire dataset, which further gets divided into two or more homogeneous sets.
- **Leaf Node:** Leaf nodes are the final output node, and the tree cannot be segregated further after getting a leaf node.
- **Splitting:** Splitting is the process of dividing the decision node/root node into sub-nodes according to the given conditions.
- **Branch/Sub Tree:** A tree formed by splitting the tree.
- **Parent/Child node:** The root node of the tree is called the parent node, and other nodes are called the child nodes.

HOW DOES THE DECISION TREE ALGORITHM WORK?

- In a decision tree, for predicting the class of the given dataset, the algorithm starts from the root node of the tree. This algorithm compares the values of the root attribute with the record (real dataset) attribute and, based on the comparison, follows the branch and jumps to the next node.
- For the next node, the algorithm again compares the attribute value with the other sub-nodes and move further. It continues the process until it reaches the leaf node of the tree.

HOW DOES THE DECISION TREE ALGORITHM WORK?

The complete process can be better understood using the below algorithm:

- Step-1: Begin the tree with the root node, says S, which contains the complete dataset.
- Step-2: Find the best attribute in the dataset using Attribute Selection Measure (ASM).
- Step-3: Divide the S into subsets that contain possible values for the best attributes.
- Step-4: Generate the decision tree node, which contains the best attribute.
- Step-5: Recursively makes new decision trees using the subsets of the dataset created in step -3. Continue this process until a stage is reached where you cannot further classify the nodes and called the final node as a leaf node.

ATTRIBUTE SELECTION MEASURES

- While implementing a Decision tree, the main issue arises that how to select the best attribute for the root node and for sub-nodes. So, to solve such problems there is a technique which is called as **Attribute selection measure or ASM.** By this measurement, we can easily select the best attribute for the nodes of the tree. There are two popular techniques for ASM, which are:
    - **Information Gain**
    - **Gini Index**

1. INFORMATION GAIN

- Information gain is the measurement of changes in entropy after the segmentation of a dataset based on an attribute.
- It calculates how much information a feature provides us about a class.
- According to the value of information gain, we split the node and build the decision tree.
- A decision tree algorithm always tries to maximize the value of information gain, and a node/attribute having the highest information gain is split first. It can be calculated using the below formula:

Information Gain= Entropy(S)- [(Weighted Avg) *Entropy(each feature)

1. INFORMATION GAIN

Entropy: Entropy is a metric to measure the impurity in a given attribute. It specifies randomness in data. Entropy can be calculated as:

```
Entropy(s)= -P(yes)log2 P(yes)- P(no) log2 P(no)
```

Where,

- **S= Total number of samples**
- **P(yes)= probability of yes**
- **P(no)= probability of no**

2. GINI INDEX

- Gini index is a measure of impurity or purity used while creating a decision tree in the CART(Classification and Regression Tree) algorithm.
- An attribute with the low Gini index should be preferred as compared to the high Gini index.

GINI INDEX

- Gini Index for a given node t :

$$GINI(t) = 1 - \sum_j [p(j \mid t)]^2$$

($p(j \mid t)$ is the relative frequency of class j at node t).

  – Maximum ($1 - 1/n_c$) when records are equally distributed among all classes, implying least interesting information
  – Minimum (0.0) when all records belong to one class, implying most interesting information

C1	0
C2	6
Gini=0.000	

C1	1
C2	5
Gini=0.278	

C1	2
C2	4
Gini=0.444	

C1	3
C2	3
Gini=0.500	

EXAMPLES FOR COMPUTING GINI

$$GINI(t) = 1 - \sum_j [p(j \mid t)]^2$$

C1	**0**
C2	**6**

$P(C1) = 0/6 = 0 \quad P(C2) = 6/6 = 1$

$Gini = 1 – P(C1)^2 – P(C2)^2 = 1 – 0 – 1 = 0$

C1	**1**
C2	**5**

$P(C1) = 1/6 \quad P(C2) = 5/6$

$Gini = 1 – (1/6)^2 – (5/6)^2 = 0.278$

C1	**2**
C2	**4**

$P(C1) = 2/6 \quad P(C2) = 4/6$

$Gini = 1 – (2/6)^2 – (4/6)^2 = 0.444$

SPLITTING BASED ON GINI

When a node p is split into k partitions (children), the quality of split is computed as,

$$GINI_{split} = \sum_{i=1}^{k} \frac{n_i}{n} GINI(i)$$

where, n_i = number of records at child i,

n = number of records at node p.

SPLITTING BASED ON GINI

$$GINI(t) = 1 - \sum_j [p(j \mid t)]^2$$

$$GINI_{split} = \sum_{i=1}^{k} \frac{n_i}{n} GINI(i)$$

Example: Marital Status

$GINI_{Msin}=1-((2/4)^2 + (2/4)^2) = 0.5$

$GINI_{Mma}=1-(0/4)^2 + (1/4)^2)= 0$

$GINI_{Mdiv}=1-(1/2)^2 + (1/2)^2)= 0.5$

$GINI_M=(4/10)*0.5 + (4/10)*0 +$

$(2/10)*0.5 = 0.45$

categorical categorical continuous class

Tid	Home Owner	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

SPLITTING BASED ON GINI

	categorical	categorical	continuous	class
Tid	**Home Owner**	**Marital Status**	**Taxable Income**	**Cheat**
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

THANK YOU